PATENT ABSTRACTS OF JAPAN

(11)Publication number : 04-096818
(43)Date of publication of application : 30.03.1992

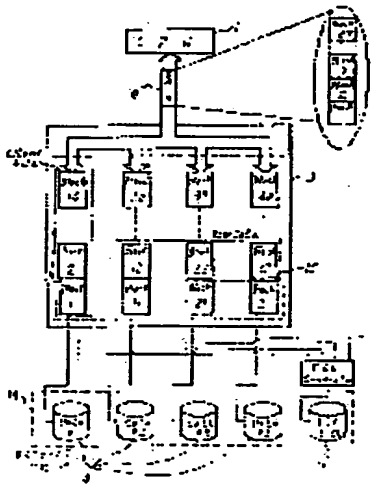
(51)Int.Cl. G06F 3/06
G06F 3/06
G11B 17/22
G11B 19/02
G11B 20/00
G11B 20/12
G11B 20/18
G11B 20/18
G11B 27/10

(21)Application number : 02-214261
(22)Date of filing : 15.08.1990

(71)Applicant : HITACHI LTD
(72)Inventor : TSUNODA HITOSHI
KAMO YOSHIHISA

(54) COLLECTIVE DISK DEVICE

(57)Abstract:
PURPOSE: To increase the transfer speed to improve the processing efficiency by writing matrix data corresponding to a disk drive in parallel and reading out data from the disk drive with one block as the unit.
CONSTITUTION: Write data transferred from a CPU 1 at a high speed in serial is distributed as matrix data 10 corresponding to an ECC disk group 14 in a data buffer 3. Data 9 and data 10 consists of many blocks. After data 9 is expanded to data 10, it is collectively written on data disks 4 in a group 14 from the buffer 3. At the time of write to disks 4, blocks are transferred to an ECC generator 6 also, and ECC is generated by blocks and is recorded on an ECC disk 5. At the time of read, only disks 4 where data is stored are accessed to read out data. Thus, the speed of transfer from the CPU is increased to perform the processing at a high speed.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平4−96818

⑤Int.Cl.⁵		識別記号		庁内整理番号
G 06 F	3/06	3 0 1	R	7232−5B
		3 0 1	N	7232−5B
		3 0 5	C	7232−5B
G 11 B	17/22			7719−5D
	19/02		F	7627−5D
	20/00		D	9197−5D
	20/12			9074−5D
	20/18	1 0 1		9074−5D
		1 0 2		9074−5D
	27/10		D	8224−5D

⑬公開　平成4年(1992)3月30日

審査請求　未請求　請求項の数 12　(全15頁)

⑭発明の名称　集合ディスク装置

⑳特　願　平2−214261

㉒出　願　平2(1990)8月15日

⑦発明者　角田　仁　東京都国分寺市東恋ケ窪1丁目280番地　株式会社日立製作所中央研究所内

⑦発明者　加茂　善久　東京都国分寺市東恋ケ窪1丁目280番地　株式会社日立製作所中央研究所内

⑦出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地

⑦代理人　弁理士　小川　勝男　外1名

明　細　書

1．発明の名称

集合ディスク装置

2．特許請求の範囲

1．上位装置からのデータの入出力要求に対する、当該データを格納するディスクドライブと、該ディスクドライブを管理する制御装置からなるシステムにおいて、上記上位装置からシリアルに転送されてきたデータを分割し、さらにそれらのデータからエラー検出、ＥＣＣを生成し、分割したデータおよびＥＣＣを別々のディスクドライブに並列で一度に格納することによりグループを作成し、読出しの場合は個々のデータのみへのアクセスを行うことを特徴とする集合ディスク装置。

2．前記集合ディスク装置において、上位装置から転送されてくる書込みデータの大きさが一定である場合、その書込みデータをあらかじめ決めておいた容量に分割し複数の領域に分けておき、その各々の領域を一端、半導体メモリに格納し、そこで、一度に並列に格納するディスクドライブの台数と同じ数に分割することにより、マトリックス状のデータに変換することを特徴とする特許請求範囲第1項記載の集合ディスク装置。

3．前記集合ディスク装置において、データ格納用ディスクドライブとＥＣＣ格納用ディスクドライブにデータ及びＥＣＣを格納する際、ECCの格納の終了を待たずに次のデータの格納を開始してしまうことを特徴とする特許請求範囲第1項記載の集合ディスク装置。

4．前記集合ディスク装置において、少なくともデータおよびＥＣＣをパラレルに書き込む時と、ディスクドライブに障害が発生し、その中のデータを復元するため、残りのデータとＥＣＣのデータを読み出す時に、当該データ格納用ディスクドライブとＥＣＣ格納用ディスクドライブの回転を同期させることを特徴とする特許請求範囲第1項記載の集合ディスク装置。

5．前記集合ディスク装置において各データ格納

用ディスクドライブにバッファを設け、ディスクドライブとバッファの双方にデータディスクに格納されているデータのカタログを格納しておくことにより、データ格納用ディスクドライブからデータを読み込む場合、そのアドレスはバッファを見ることにより行なうことを特徴とする特許請求範囲第1項記載の集合ディスク装置。

6．前記集合ディスク装置において上位装置からシリアルに転送されてきたデータを、マトリックス状のデータに変換しパラレルに格納するディスクドライブ数に分割した際、当該ディスクドライブに格納されるデータの長さが揃わない場合不足している分だけ、ある特殊なデータを埋め込み、長さを揃えることを特徴とする特許請求範囲第1項記載の集合ディスク装置。

7．前記集合ディスク装置において上位装置からシリアルに転送されてきたデータを、パラレルに格納するディスクドライブに、ある単位で順に振り分け、マトリックス状のデータを作成しデータがそろい次第パラレルに転送することを特徴とする特許請求範囲第1項記載の集合ディスク装置。

8．前記集合ディスク装置においてを当該データ格納用ディスクドライブにより構成されるグループを論理的なボリュームとして扱うことを特徴とする特許請求範囲第1項記載の集合ディスク装置。

9．前記集合ディスク装置において、一括書き込みを行っている際当該ディスクドライブ内の当該トラックが書き込み不能の場合、そのトラックへの書き込みは行わず、又、該ディスク上の予備のトラックへヘッドを移動して書き込む動作を行わずに、そのトラックをとばして、同一シリンダ内の他のトラックへ、ヘッドの電気的な切り替えにより、次のトラックへ書き込むことを特徴とする特許請求範囲第1項記載の集合ディスク装置。

10．前記集合ディスク装置において、一括書き込みを行っている際、当該ディスクドライブにより構成されるグループにおいて、一つのディスクドライブのシリンダへの書き込みが終了した時点でグループ内の他のディスクドライブへの書き込みも終了として、グループ内の全てのディスクドライブにおいてシリンダを移動することを特徴とする特許請求範囲第1項記載の集合ディスク装置。

11．前記集合ディスク装置において、複数のディスクドライブにより構成される一括書き込みを行う単位であるグループにおいて、そのグループ内の個々のディスクドライブの中で、少なくとも一台でも使用されている場合、“ＯＮ”とするグループの使用中を示すフラグを管理テーブルとして、集合ディスク装置を制御する部分に設け、一括書き込みを行う場合そのフラグを見ることにより書き込み可能かを判定することを特徴とする特許請求範囲第1項記載の集合ディスク装置。

12．前記集合ディスク装置において、複数のディスクドライブにより構成される一括書き込みを行う単位であるグループにおいて、そのグループ内の個々のディスクドライブについて使用されている場合、“ＯＮ”とする使用中フラグと、そのグループ内のディスクドライブが少なくとも一つでも使用されている場合“ＯＮ”とする使用中フラグを、階層的な管理テーブルとして集合ディスク装置を制御する部分に設けることを特徴とする特許請求範囲第1項記載の集合ディスク装置。

3．発明の詳細な説明

〔産業上の利用分野〕

本発明はコンピュータシステムに係り、特に高性能な入出力動作を可能とするディスクファイルシステムに関する。

〔従来の技術〕

現在のコンピュータシステムにおいては、CPU等の上位側が必要とするデータは２次記憶装置に格納され、ＣＰＵが必要とする時に応じて２次記憶装置に対してデータの書込み，読みだしを行っている。この２次記憶装置としては一般に不揮発

な記憶媒体が使用され、代表的なものとして磁気ディスク、光ディスクなどが使用されている。特許，学術文献等のデータベースなどでは、通常のユーザが使用する場合は２次記憶装置から読みだすのみで、ユーザが直接データを書換えることはない。データの変更，追加を行う場合は、データベースを管理する者が一度にまとめて追加という形で行う。

従来、多数の比較的容量の小さな磁気ディスクを用意し、ＣＰＵから転送されてくるデータを分割し、複数の磁気ディスクに同時に格納し、読み出す場合は逆に各々の磁気ディスクから同時に読み込んできてＣＰＵへ転送するパラレル処理を行うという技術がある。このようなシステムについてはストリッジ　コンセプト社等の企業から製品発表等されている。また、情報処理学会第３９回全国大会において桜井　紀彦らが"アレイ制御ディスクの構成方式に関する一考察"においてデータを並列に処理したり、通常のようにシリアルなデータとして処理したりを一つのシステム内において

から転送されてきた一つのデータをｎ個に分割しデータディスク＃１から＃ｎに並列に一度に書き込み、読み出す際は逆にデータディスク＃１から＃ｎより一度に読み込む方式が特開平１－２５０１２８に開示されている。しかし、このように一括に書き込み、一括に読みだす方式では、大量のデータをまとめて扱う場合は良いが、小さなデータを読み出す場合、データディスク＃１から＃ｎを全て使用するためシステムの処理効率が悪い。

〔課題を解決するための手段〕

上記問題を解決するため一括にデータを書き込むときは上位装置であるＣＰＵからシリアルに転送されてきた大量のデータを、データを読み出す際のアクセス単位であるブロックを基準として、マトリックス状のデータ構成に変換し、各当該データ格納用ディスクドライブに対応したマトリックスデータ内のデータをパラレルに書き込む。また、読み出す際は、個々のディスクドライブからブロックを単位として読み出すことで解決できる。

〔作用〕

いてダイナミックに切り替えて行う方式について報告している。

〔発明が解決しようとする課題〕

特許、学術文献等のデータベースなどでは、通常のユーザが使用する場合は２次記憶装置から個々のデータを読みだすのみで、ユーザが直接データを書換えるようなことはない。データの変更、追加を行う場合は、データベースを管理する者が一度にまとめて追加という形で行う。この追加作業はＣＰＵから送られてくる大量のシリアルなデータを、それを構成する個々のアクセス単位であるブロックについて細かく、しかも順次行われるため、書込み時間が大量にかかる。また、上記したようにシリアルに転送されてくるデータ内の個々のデータについて行うことから、磁気ディスクの処理時間により上位装置であるＣＰＵの転送速度が決まってしまうため、上位装置であるＣＰＵからの転送速度をあげて高速に処理することができない。以上のような課題を解決する方法として、第１図のようなディスクドライブの構成で、CPUI

本発明のようにマトリックスデータを作成してパラレルに書込み処理を行うことにより個々のディスクドライブの転送速度は同じでもシステムとしての転送速度が上がる。また、磁気ディスクにおける処理時間が短縮されるため、上位装置であるＣＰＵからの転送速度をあげられる。一方読出し時では個々のディスクドライブへのアクセスが可能となるため、処理効率が向上する。

〔実施例〕

実施例１

以下本発明の一実施例を第１図により説明する。本実施例はＣＰＵ１，データ制御部(以下ＤＣＵ)２，データバッファ（以下バッファ）３，データ用磁気ディスク４(以下データディスク)，ＥＣＣ用磁気ディスク(以下ＥＣＣディスク)５，ＥＣＣジェネレータ６，Read用データバッファ（以下Readバッファ）７，データ復元部８より構成される。ＣＰＵ１より発行された１／Ｏ要求はＤＣＵ２を通して各磁気ディスクに発行される。

まず、本実施例における具体的な１／Ｏ処理に

ついての大まかな動作を第１図及び第２図を用いて簡単に説明する。本実施例ではデータの記憶又は書き替えはＥＣＣを作成するグループ単位（これをＥＣＣディスクグループ１４という）で一度に大量に行う。ＣＰＵ１より高速で、シリアルに転送されてきた書込みデータ９は第１図のDCU2の管理のもとでデータバッファ３内においてＥＣＣディスクグループ１４に対応したマトリックスデータ１０として振り分けられる。なお、書込みデータ９及びマトリックスデータ１０は多数のブロックにより構成されている。このブロックという単位は上位からの１アクセス単位である。つまり、上位（ＣＰＵ１）から、データを読み込めというRead要求が当該データディスク４へ発行された場合、当該データディスク４からReadする単位は１ブロックである。このように、書込みデータ９をマトリックスデータ１０に展開した後、バッファ３より各ブロックを当該データディスクグループ１４内の各データディスク４にまとめて書き込む。また、データディスク４に書き込む際、各々のブロックはＥＣＣジェネレータ６にも同時に転送され、各々のブロックからＥＣＣを作成しＥＣＣディスク５に同様に記録される。読出し時はＥＣＣディスクグループ１４内における当該データが格納されている当該データディスク４のみにアクセスし当該データを読み出す。本実施例では以上のようにＩ／Ｏを処理する。

そこで、以下にＩ／Ｏ処理動作について詳細に説明する。

上記したように本実施例では、一括してデータを書き込むため、シリアルなデータとしてＣＰＵ１より転送されてきた書込みデータ９はマトリックスデータ１０に変換する必要がある。そこで、まずこの変換方法について第３図により説明する。本実施例では各データディスク４及びＥＣＣディスク５のトラック容量（磁気ディスクの一周に格納できるデータの容量）は４０ＫＢであり、各ブロックのデータ量は４ＫＢで、データディスク４の台数は４台である。第３図に示すようにＣＰＵ１からの書込みデータはシリアルにまとめて、大量に送られてくる。この、まとめて送られてくる書込みデータ９の量は、常に一定である方が望ましい。この書込みデータ９の容量はあらかじめＤＣＵ２に登録されている。次に、大量の書込みデータ９は既に前もって登録されている容量（読込み時の１アクセス単位であるブロックの整数倍）に区切られ、ｎ個の領域に分けられる。次に、各領域は、データディスク４の台数分にブロックを単位として区切られる。この４分割されたデータが、それぞれのデータディスク４に格納されるブロックの集まりとなりColumnデータと呼ぶ。以上のようにＤＣＵ２の管理のもとでデータバッファ３内において各データディスク４に格納するブロックの集まり（Columnデータ）を、各データディスク４に対応して作成した物をマトリックスデータ１０とする。本実施例では書き込む際はＥＣＣディスクグループ１４内の４台のデータディスク４を１つの論理ボリュームとして考える。つまり、論理的なトラック容量は物理的なトラック容量（個々のデータディスク４及びＥＣＣディスク５のトラック容量）のデータディスク４の台数倍となる。

次に、このようにして作成したマトリックスデータ１０を、一括に書き込む方法について第４図を用いて説明する。第４図はデータディスク４及びＥＣＣディスク５の内部構成を示す。データディスク４及びＥＣＣディスク５はｎ枚のディスクと、ヘッドを移動させるアクチュエータ１１に、ｎ個のヘッドがついて構成される。各々のヘッドは１つのアクチュエータ１１についており、アクチュエータ１１が移動することにより、全てのヘッドが同じように動く。つまり、アクチェータ１１の動きによりｎ個のヘッドに対応するｎ個のディスク上のトラックが決定される。このトラックの集合のことをシリンダという。磁気ディスクではディスクに同心円状にトラックが設定されているため、外周から内周に向いトラックにアドレスを割り当てる。このアドレスのことをシリンダナンバという。そこで、Ｒ／Ｗ回路１２は当該ヘッドを選択してトラックを決定しデータの読み書

きを行なう。各ディスク１周の容量は物理的なトラック容量である。このような構成のデータディスク４、ＥＣＣディスク５によりＥＣＣディスクグループ１４が構成される。まず、領域１を書き込む場合、第３図に示すような領域１に対応したマトリックスデータ１０を作成し、各Columnデータを、当該データディスク４に書き込む。第４図に示すように各々の当該データディスク４ではヘッド＃１を選択してディスク＃１の斜線の部分に第３図における領域１のマトリックスデータ１０内の当該データを書き込む。なお、ヘッド＃１，２を同時にＲ／Ｗできる場合は、同時に行う。ディスク＃１に書込みが終了し領域１のデータの書込みが終了していない場合、ヘッド＃２を電気的な切り替えにより選択し、ディスク＃１と同様に、ディスク＃２の斜線の部分に書き込む。以下ディスク＃３，＃４……＃ｎと同様に書き込む。このようにして、領域２，３，４，５……ｎと同様に書き込んでいく。なお、ディスク＃２上のトラックが、ディスク面上の欠陥等により使用できない場合、現在はあらかじめ用意してある予備のトラックへヘッドを移動させ使用している。本発明ではそのような予備のトラックへの切り替えを無視し、ディスク＃２をとばしてディスク＃３に書き込む。このように、各領域をシリングに書き込んでいき、ＥＣＣディスクグループ１４内のデータディスク４のうち、一つでもディスク＃ｎを書き終えた時点で、そのシリンダへの書き込みを終了とし、別のシリンダに移って書き込む。

一方、このように各データディスク４にデータを書き込むと同時に、データをＥＣＣジェネレータ６にも転送し、ＥＣＣを作成し、ＥＣＣディスク５に書き込む。そこで、以下にＥＣＣ作成方法について説明する。ＥＣＣ作成方法は、書込みデータ９をマトリックスデータ１０に変換し、第７図に示すように各データディスク４に対応したColumnデータに展開する。そこで、マトリックスデータ１０についてパリティ作成グループ１３（Rowデータ）においてＥＣＣジェネレータ６によりＥＣＣ（例えば奇数パリティ）を作成する。

このように一括書込みの際にマトリックスデータ１０に対してＥＣＣを作成することにより、もしある１台のデータディスク４に障害が発生し、対応したColumnデータが読込み不能になった場合、残りのデータとＥＣＣから障害データディスク４に対応したColumnデータを復元することが可能となる。

このように各領域においてマトリックスデータ１０を作成し、マトリックスデータ１０内のそれぞれのデータディスク４に対応したColumnデータを並列に書き込むことにより高速な処理が可能となる。

以上マトリックスデータ１０の作成方法及び格納方法について説明したが、第９図に示すようにＣＰＵ１からの書込みデータ９を領域に区切っていき、最後の領域ｎにおいてマトリックスデータ１０を作成した時、各Columnデータ（各データディスク４に対応して格納されるブロックの集合）の長さが揃わない場合がある。そこで、このような場合の処理について以下に示す。ＤＣＵ２はマトリックスデータ１０を作成した時、第９図に示すように、各Columnデータの長さが揃っていないことを認識している。第９図のマトリックスデータ１０を見るとデータディスク＃４に対応したColumnデータは他のColumnデータと比較し長さが２ブロック分短くなっている。そこで、このようなマトリックスデータ１０を各データディスク＃４に並列に書き込んでいく場合、データディスク＃４に対応したColumnデータはブロック３８までの転送を終了した時点で、そのColumnデータの転送を終了とする。この時、すなわち各Columnデータの長さが揃わないことが生じた場合のＥＣＣ作成方法について説明する。ＥＣＣの作成においては、第９図のマトリックスデータ１０のデータディスク＃４に対応したColumnデータで、ブロック３１から３８までのRowデータはデータディスク＃１から＃４に対応したColumnデータを並列にＥＣＣジェネレータ６に転送し、それ以降はデータディスク＃１から＃３のみを並列に転送してＥＣＣを作成する。以上のような並列に転送する

データディスク４の台数をダイナミックに切り替えるのはＤＣＵ２が制御する。

以上のようにマトリックスデータ１０を作成し、データ及びＥＣＣを当該ＥＣＣディスクグループ１４に格納するのだが、その時の書き込むタイミングについて、第１６図により説明する。マトリックスデータ１０内の各Columnデータを当該ECCグループ１４内の各データディスク４に並列にデータを転送し、同時に、ＥＣＣジェネレータ６にも同様にデータを転送し、ＥＣＣを作成する。しかし、ＥＣＣの作成にはある一定の処理時間が必要であり、データディスク３へのデータ格納開始時間とＥＣＣディスク５へのデータ格納開始時間との間にずれが生じる。この時、各データディスク４及びＥＣＣディスク５への記録タイミングには、次の同期、非同期の２方式が考えられる。同期した場合はデータの格納とＥＣＣの格納はセットで考えられ、次の領域の格納は、ＥＣＣの格納が終了した時点で開始される。一方、非同期の場合はデータの格納とＥＣＣの格納は別と考え、ＥＣＣの格納の終了を待たずに次の領域のデータの格納を開始する。第１６図にデータ格納とECC格納を同期して扱った場合と、非同期に扱った場合の書込み時間についてしめす。この図から、同期した場合、領域１，２を格納したとき

格納時間＝データ書込み時間＋ＥＣＣ作成のずれ×２

となる。もしｎ個の領域まで格納したとすると

格納時間＝データ書込み時間＋ＥＣＣ作成のずれ×ｎ

となる。一方非同期で領域１，２を格納した場合

格納時間＝データ書込み時間＋ＥＣＣ作成のずれ

となり、これはｎ個の領域を格納した場合も同じである。以上のことからデータの書込み動作とＥＣＣの書込み動作を非同期に扱う方が書込みデータ９を一括で書き込む際に必要となる時間が短くなる。以上のことから、データとＥＣＣの格納を非同期に扱った方が望ましい。

このようにして、データ及びＥＣＣを格納する際、ＥＣＣディスクグループ１４内の各データディスク４，ＥＣＣディスク５の回転制御について次に示す。ＥＣＣディスクグループ１４内の各データディスク４，ＥＣＣディスク５の回転を同期させることにより、まとめて書きこむ場合の平均回転待ち時間（Read／Vrite ヘッドの下に当該レコードが到達するまでのディスクの回転時間）は、１台の時と同じ１／２回転となる。このことから、各データディスク４，ＥＣＣディスク５は回転を同期させて書き込むことが望ましい。

次に、データディスク４内の個々のデータを読み出す場合を第２図を用いて以下に示す。本実施例では、ブロック（読込み時のアクセス単位）の大きさは４ＫＢである。読込み時ではＥＣＣディスクグループ単位でのアクセスではなく、当該レコードを格納している、当該データディスク４へのアクセスとなる。

ＣＰＵ１から発行されたRead要求はＤＣＵ２においてコマンド解読され、アドレス変換を行なった後、ＤＣＵ２の管理のもとで当該データドライブ４に読込み要求を発行する。当該データドライブ４ではＤＣＵ２からのコマンドを受け付け次第ＣＰＵ１との接続を切り、他のデータディスク４でのRead処理にＤＣＵ２及びパスを空けわたす。当該データディスク４ではＤＣＵ２及びパスをあけ渡した後、独自で当該レコードが格納されているトラックまでヘッドを移動させるシーク及び当該レコードがヘッドの下に来るまで回転待ちを行なう。シーク及び回転待ちを行なった後、当該レコードの読込みが可能になり次第、ＤＣＵ２およびパスを再び獲得しRead用Data Buffer（ＲＤＢ）７にデータを転送し、そこからデータを高速でＣＰＵ１に転送する。なお、読みだし時はデータディスク４、ＥＣＣディスク５の回転を同期しても、しなくても構わない。

次にアドレスの管理方法について説明する。

まず、書込み時について説明する。書込み時、ＣＰＵ１はＥＣＣディスクグループ１４を論理ボリュームとする。ＤＣＵ２内にはＥＣＣディスクグループと、そのグループ内の個々のデータディスク４、ＥＣＣディスク５の使用状況を示す管理テーブルを持っている。このＥＣＣディスクグループの管理テーブルと、そのグループ内の個々の

データディスク４，ＥＣＣディスク５の管理テーブルは階層的な構造になっている。ＥＣＣディスクグループ内のデータディスク４とＥＣＣディスク５の中で１台でも使用されている場合、データディスク４及びＥＣＣディスク５の使用状況を示す管理テーブル内の該当した使用中であることを示すフラグが“ＯＮ”（使用中）になり、同時に所属するＥＣＣディスクグループが使用中であることを示すフラグが“ＯＮ”（使用中）になる。書き込みに際しＣＰＵ１は第１９図に示すようにＥＣＣディスクグループの使用状況を示す管理テーブルを見て、当該ＥＣＣディスクグループ１４が使用中であるかフラグを調べる。前のＩ／Ｏ要求により当該ＥＣＣディスクグループ１４が使用中の場合は再度Ｉ／Ｏ要求を発行し、使用されていなければ、書込み処理に入る。このようなフラグを用意しておくことにより、一括書き込みの際当該ＥＣＣディスクグループ１４内の各々のデータディスク４について使用状況をチェックせずにすむ。また、本実施例におけるシステムのトラック容量は個々のデータディスク４の実際のトラック容量の４倍（ＥＣＣグループ内のデータディスク数が４台の場合）とする。本実施例では第２図に示すようにＣＰＵ１からはシーケンシャルな書込みデータ９として送られてくる。この、書込みデータ９を当該ＥＣＣグループ１４に格納する場合、その書込みデータ９はＤＣＵ２の管理のもとで、データバッファ３内においてマトリックスデータ１０に変換され、ＥＣＣディスクグループ１４内の各データディスク４に並列同時に書き込まれる。この時同時に以下に示すようにアドレス変換テーブル及び、各データディスク４内のカタログに書き込む。通常、書込み時はＣＰＵ１が書込みデータ９に対し、データセット名を割り当てて転送する。そこで、ＤＣＵ２ではＣＰＵ１（ＯＳ）が持つデータセット名とＤＣＵ２が実際に管理するアドレス（ＥＣＣグループナンバ，データディスクナンバ）との対応表として、アドレス変換テーブルをＤＣＵ２内部のメモリに格納する。この、アドレス変換テーブルを格納するメモリは揮発でも良いが、不揮発が望ましい。さらに、このアドレス変換テーブルはアドレス変換テーブル用ディスク１５にも同時に書き込まれ、信頼度を高めることが望ましい。また、書込みデータ９はマトリックスデータ１０状に展開され、個々のColumnデータとして当該データディスク４に格納されるため、それぞれのデータディスク４において格納されるデータのヘッドナンバ、シリンダナンバ、レコードナンバを当該データディスク４内のカタログに記録する。

一方読込み時ではＣＰＵ１（ＯＳ）からは当該データセット名を指定し、読み込み可能かを調べる。第１９図に示すように、まず、当該ＥＣＣディスクグループ１４が使用中であることを示すフラグが“ＯＦＦ”（使用されていない）の場合はそのまま読込みにかかり、“ＯＮ”（使用中）の場合は当該データディスク４が使用中かを調べる。使用可能の場合、第１７図に示すようにＣＰＵ１より指定されたデータセット名から、ＤＣＵ２がそのデータセットが格納されているデータディスク４のアドレスを割り当てる。この割り当てる方法について以下に説明する。第１８図に示すようにＣＰＵ１から指定されたデータセット名から当該ＥＣＣディスクグループ１４を見付ける。第１７図に示すように、そのアドレス変換テーブルにおいて当該データセットが格納されているアドレスがａからｂまでならデータディスク＃１に対応し、ｂからｃならデータディスク＃２、ｃからｄならデータディスク＃３、ｄからｅならデータディスク＃４に対応する。このように、当該データセットが格納されている、当該データディスク４のアドレスを見つけ、次に、第５図に示すように、そのデータディスク４内のカタログにより当該データが格納されているヘッドナンバ，シリンダナンバ，レコードナンバを調べ、当該データを読み出す。なお、上記アドレス管理方法においては第５図に示すようにカタログを各データディスク４内に置いたが、高速なアクセスを要求する場合は、このカタログをＤＣＵ２内のアドレス変換テーブルに置く方が望ましい。このようにするこ

とにより、ＣＰＵ１から転送されてきたデータセット名に対し、即座にデータディスクナンバ，ヘッドナンバ，シリンダナンバ，レコードナンバを割当て、高速に当該データをアクセスすることが可能となる。また、各データディスク４内のみにカタログを置くのではなく、各データディスク４にカタログ用のバッファを設け、いちいちカタログを読みにデータディスク４をアクセスせずに、高速に当該データのアドレスを見つけることも可能となる。

以上述べてきたように、本実施例を用いることにより、書込みデータ９をマトリックスデータ１０に変換する時の変換アルゴリズムは簡単になる。また、個々のブロックの管理も楽である。

本実施例では、トラック容量，ブロックのデータ量，データディスクの台数を変えた場合でも、同様な効果が期待できることは明らかである。また、データディスク４及びＥＣＣディスク５を磁気ディスクとしたが、光ディスクディスク，フロッピーディスクなどを使用しても同様なことが可能である。

実施例２

ＤＣＵ２がマトリックスデータ１０を作成した時、各Columnデータの長さが揃っていない場合の処理方法の他の実施例を実施例２として以下に示す。第９図でＣＰＵ１からの書込みデータ９を領域に区切っていき、最後の領域ｎにおいてマトリックスデータ１０を作成した時、各Columnデータ（各データディスク４に対応して格納されるブロックの集合）の長さが揃わない場合がある。本実施例でも実施例１と同様に一括書込みの際に第７図に示すようにＥＣＣを作成し、障害発生時は残りのデータとＥＣＣより障害データディスク４に対応したColumnデータを復元することが可能となる。そこで、このような場合のＥＣＣ作成方法について次に説明する；ＤＣＵ２はマトリックスデータ１０を作成した時、第９図に示すように、各Columnデータの長さが揃っていないことを認識している。

そこで、本実施例では、Columnデータの長さを揃える目的で、第６，８図に示すように、不足しているデータ分だけ“０”あるいは任意のデータを書き込んでおく。このようにして、見かけ上各Columnデータの長さが等しいマトリックスデータ１０を作成し、通常のＥＣＣ作成アルゴリズムによりＥＣＣを作成する。以上のように、ＤＣＵ２によりマトリックスデータ１０の各Columnの長さがたりない部分にＡＬＬ０のデータを書き込み、長さを揃えた方が、ＥＣＣ作成アルゴリズム内に、長さが揃わない場合のアルゴリズムを用意する必要がないため、単純化されて良い。

なお、本実施例では長さがたりない分、“０”を書き込むことにしたが、“０”以外の特定のパターンを書き込んでも良い。

また、本実施例は実施例１に記載の他のすべての動作に適応可能である。

実施例３

以下にマトリックスデータ作成方法についての他の実施例を実施例３として示す。本実施例ではまとめて一括書込みを行う際、書込みデータ９を、Row 方向にデータを作っていき（当該ＥＣＣディスクグループ１４の各データディスク４に順番に振り分けていく）マトリックスデータ１０に変換する方法である。本実施例では実施例１と同様、各データディスク４及びＥＣＣディスク５のトラック容量(磁気ディスクの一周の容量)を４０ＫＢ、各ブロックのデータ量を４ＫＢとしデータディスク４の台数を４台する。

第１０図に示すようにＣＰＵ１よりまとめて大量に転送されてきた書込みデータ９はＤＣＵ２の管理のもとでマトリックスデータ１０に変換される。まず、第１１図に示すように、実施例１と同様に書込みデータ９はｎ個の領域に分けられる。次に、各領域は、データディスク４の台数分に振り分けられる。この振り分けられたデータが、それぞれのデータディスク４に格納されるブロックの集まりとなる。以下に振り分ける方法について第１２図を用いて具体的に説明する。ＣＰＵ１より転送されてきた書込みデータ９は多数のブロックにより構成され、シリアルに転送されてくる。

この書込みデータ９内で１個以上のブロックの集合（この集合の大きさは等しくする）を作り、書き込むデータディスク４に対応したColumnデータとして振り分けていく。書込みデータ９でBlock １からBlock Ｌまでを第１図のデータディスク＃１に格納すべくColumnデータ＃１に振り分け、Block Ｌ＋１からBlock ｍまでをデータディスク＃２に格納すべくColumnデータ＃２に振り分ける。同様にColumnデータ＃３，＃４にBlock ｍ＋１からｎ，ｎ＋１からｐを振り分ける。本実施例ではデータディスクの台数を４台としたため、Columnデータ＃４に割当てた後は再びColumnデータ＃１にBlock ｐ＋１からｑを割当て以下同様に書込みデータ９をColumnデータに展開していく。以上のようにして、書込みデータ９をマトリックスデータ１０に変換する。

このようにマトリックスデータ１０を作成することにより、一括書込みで並列にしかも高速に書き込むことが可能となる。

次に、この変換方法を使ったとき、マトリックスデータ１０内において各Columnデータの長さが揃わない場合の処理方法について述べる。第１３図のように書込みデータ９からマトリックスデータ１０を作成した時、各Columnデータの長さが揃わない場合が生じる。このような場合のデータ書込み、ＥＣＣ作成処理方法は以下のようにすれば良い。本実施例では実施例１と同様に、一括書込みの際に第７図に示すようにＥＣＣを作成する。ＤＣＵ２はマトリックスデータ１０の作成時に、第９図に示すように、各Columnデータの長さが揃っていないことをＤＣＵ２は認識している。第１３図のマトリックスデータ１０を見るとデータディスク＃３，４に対応したColumnデータは他のColumnデータと比較し長さが１ブロック分短くなっている。そこで、このようなマトリックスデータ１０を各データディスク４に並列に書き込んでいく場合、データディスク＃３，４に対応したColumnデータはブロック３５，３６までの転送を終了した時点で、そのColumnデータの転送を終了とする。ＥＣＣの作成においては、第１３図のマトリックスデータ１０のデータディスク＃３，４に対応したColumnデータで、ブロック３から３５，４から３６までのRow データはデータディスク＃１から＃４に対応したColumnデータを並列にＥＣＣ ジェネレータ６に転送し、それ以降はデータディスク＃１と＃２のみを並列に転送してＥＣＣを作成する。以上のような並列に転送するデータディスク４の台数をダイナミックに切り替えるのはＤＣＵ２が行うものとする。

次に、実施例２と同様にColumnデータの長さを揃える目的で、第１４図に示すように、不足しているデータ分だけ“０”あるいは任意のデータを書き込んでおく方法について説明する。以上のように、ＤＣＵ２によりマトリックスデータ１０の各Columnの長さがたりない部分にＡＬＬ“０”あるいは任意のデータを書き込み、見かけ上各Columnデータの長さが等しいマトリックスデータ１０を作成し、通常のＥＣＣ作成アルゴリズムによりＥＣＣを作成する。このように長さを揃えることで、ＥＣＣ作成アルゴリズム内に、長さが揃わない場合のアルゴリズムを用意する必要がなく、単純化される。

次に、本発明を適応しない場合と実施例１の書込みデータ９を一端データバッファ３に格納した後、４等分して当該ＥＣＣディスクグループ１４の各データディスク４に書き込む方法と、書込みデータ９をデータバッファ３に書き込みながら当該ＥＣＣディスクグループ１４の各データディスク４に書き込む方法において、各々の書込み時間の比較を第１５図を用いて説明する。第１５図に示したようにデータディスク４の転送レートを３ＭＢ／ｓ、回転数を３６００r.p.m、トラック容量を４０ＫＢ／ＴＲＫとし、ＥＣＣディスクグループ１４内にはデータディスク４を４台持っているものとする。ＥＣＣディスクグループへの転送データ量を

４０ＫＢ／ＴＲＫ×４台＝１６０ＭＢ／Group

とした場合、このデータを格納する場合本発明を適応しないと４回転ディスクが回転しなければならない。この時の書き込み時間は

ディスク1回転にかかる時間(16.6ms)×4回転=66.4msとなる。実施例1の方法ではデータバッファ3への書込み転送レートが12MB/sではトータルの書込み時間（160MB）は30msとなり、本発明を適応しない場合と比較し45%に減少する。また、データバッファ3への書込み転送レートを24，48MB/sとしていくと、トータルの書込み時間はさらに、23.3，20ms と減少する。一方、本実施例のようにデータバッファ3に書き込みながら、当該ECCディスクグループ14の各データディスク4に書き込む場合は、データバッファ3への書込み転送レートを12MB/s以上にすると、データバッファ3への書込み時間は、データディスク4への書込み時間中に行われるので全書込み時間は、データディスク4への書込み時間となり16.6ms となり本発明を適応しない場合と比較し25%に減少する。以上のことから本発明を適応することにより書き込み時間は減少し、また、データバッファ3への書込み転送レートが低い場合は本実施例が有効である。

なお、トラック容量，ブロックのデータ量，データディスクの台数を変えた場合でも、本実施例で示すようなマトリックスデータ作成方法を同様に使用できることは明らかである。また、本実施例は実施例1，2に記載の他のすべての動作に適応可能である。

〔発明の効果〕

以上説明したように、一括書込み，個別読み出しするような、データベースなどでは、本発明のように書込みをまとめて並列に高速で行うことにより、書込み時間を短縮することができる。さらに、一括書込み時に領域に分割しマトリックス状のデータに変換して並列に書き込む。その際、各データ書込み用磁気ディスクに対応した、データの長さが揃わない場合、その差を詰めないで使用することにより、高速に処理することが可能となる。

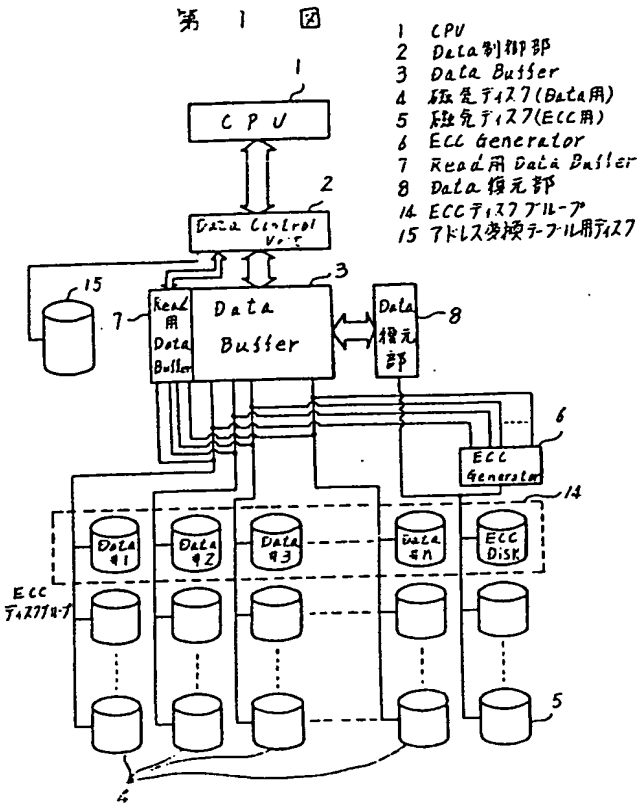
4．図面の簡単な説明

第1図は本発明の全体構成ブロック図，第2図は第1の実施例のデータ処理動作説明のためのブロック図，第3図は本発明の第1の実施例のデータ変換概念図，第4図は本発明のデータディスク，ECCディスクの内部構成図，第5図は本発明のアドレス変換の概念図，第6図は第1の実施例のデータ変換概念図，第7図は本発明のECC作成概念図，第8図は第2の実施例のECC作成の説明のための概念図，第9図は第1の実施例のデータ変換の説明のための概念図，第10図は第3の実施例のデータ処理動作説明のためのブロック図，第11図は第3の実施例のデータ変換説明のための概念図，第12図は第3の実施例のデータ変換詳細を示す概念図，第13図は第3の実施例のデータ変換を説明する概念図，第14図は第3の実施例のデータ変換を説明する概念図，第15図は本発明の書込み処理時間比較図，第16図は本発明のデータ書込み時間の比較タイムチャート，第17図は本発明のアドレス変換方法を説明するブロック図，第18図は本発明のアドレス決定方法を説明するブロック図，第19図は本発明の管理テーブルへのアクセスフローチャート。

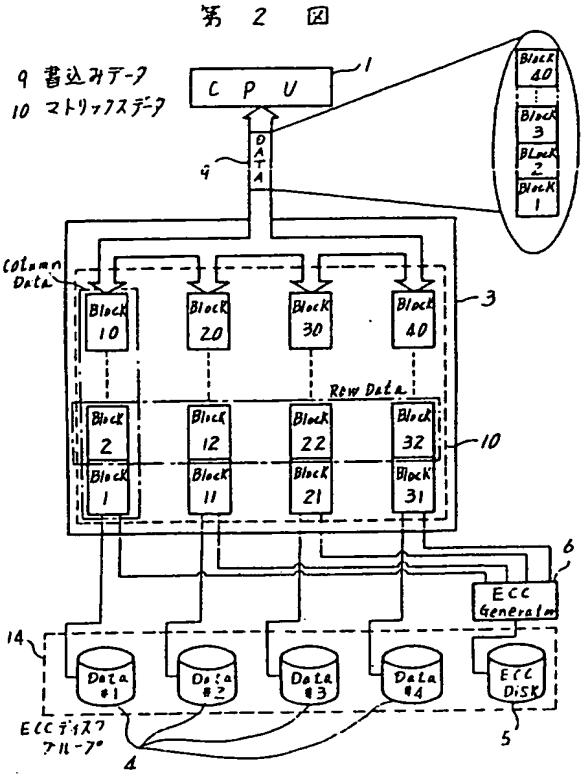
1…CPU（中央処理装置），2…DCU（Data制御部），3…データバッファ，4…データディスク（データ用磁気ディスク），5…ECCディスク（ECC用磁気ディスク），6…ECCジェネレータ，7…Read用データバッファ，8…データ復元部，9…書込みデータ，10…マトリックスデータ，11…アリチュエータ，12…R/W回路，13…パリティ作成グループ，14…ECCディスクグループ，15…アドレス変換テーブル用ディスク。

代理人　弁理士　小川勝男

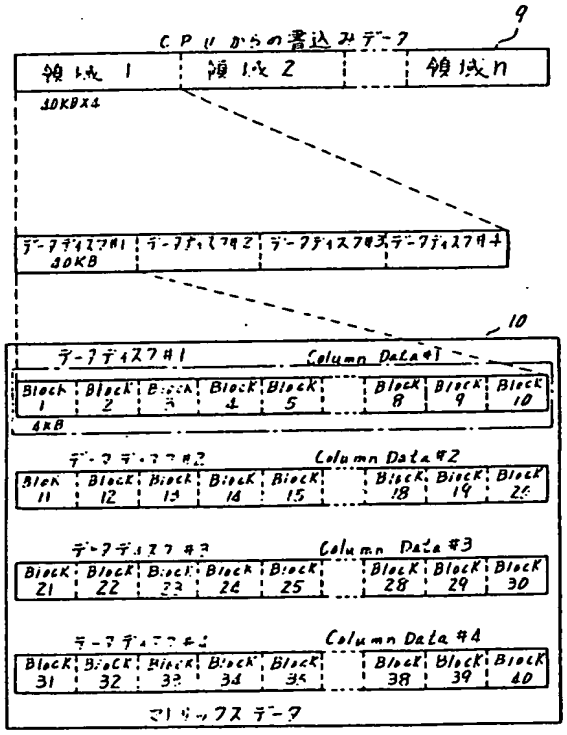
第 1 図



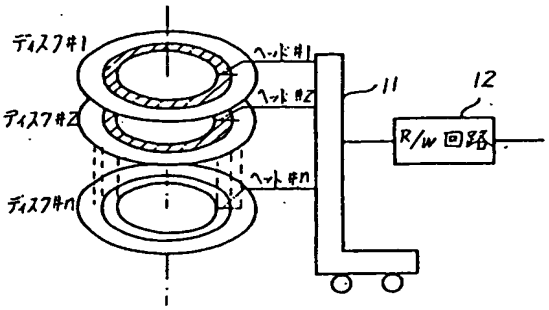
第 2 図



第 3 図

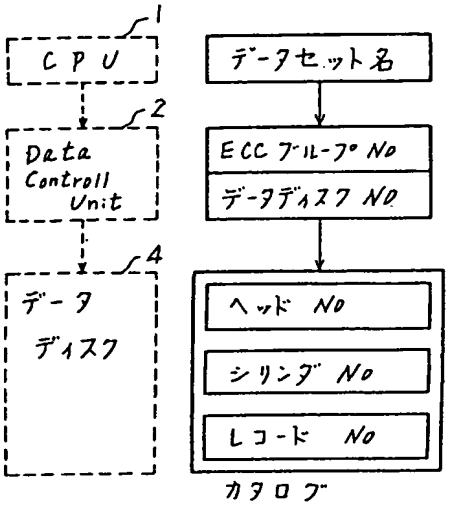


第 4 図

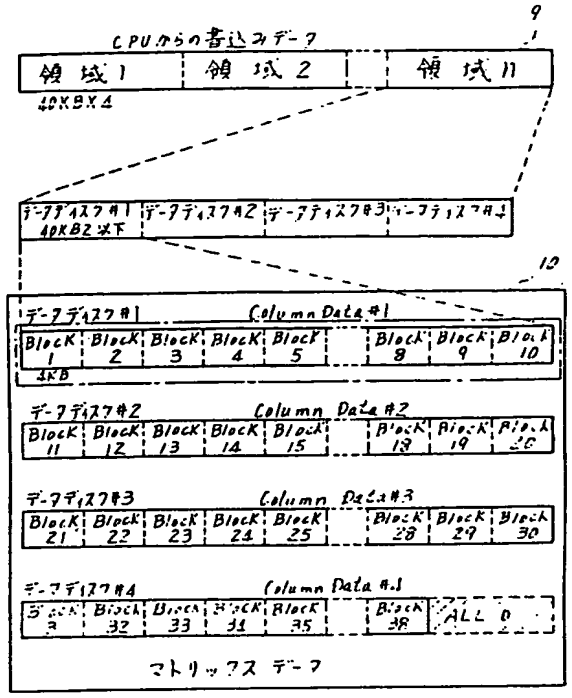


11 アクチュエータ
12 R/W回路

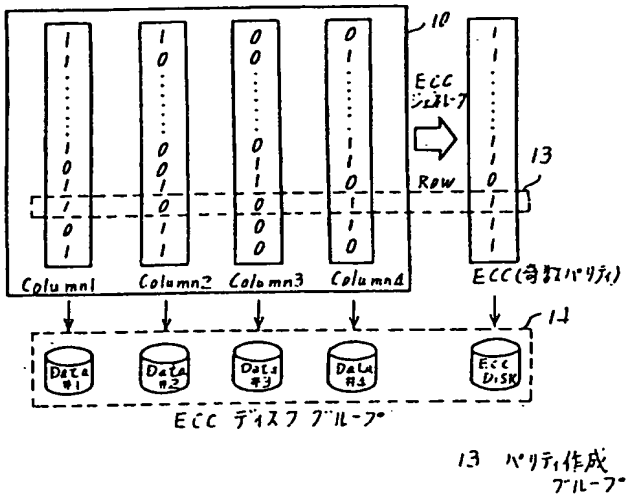
第 5 図



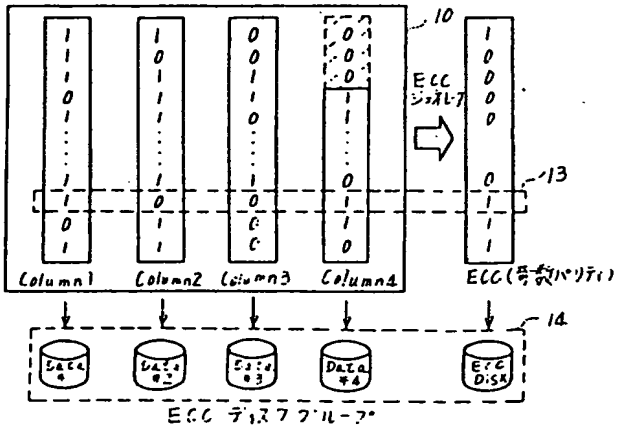
第 6 図



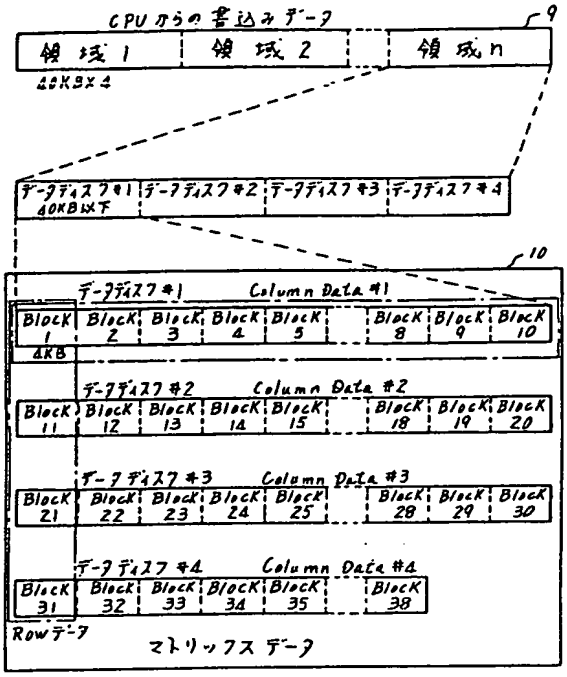
第 7 図



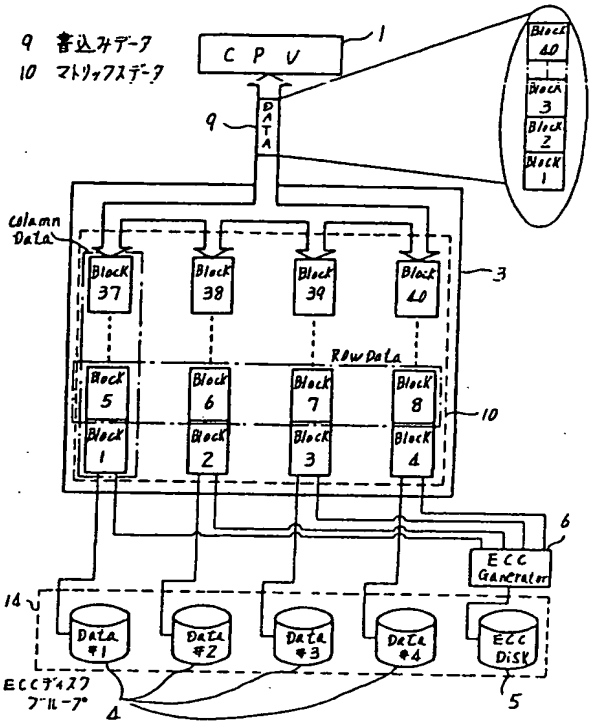
第 8 図



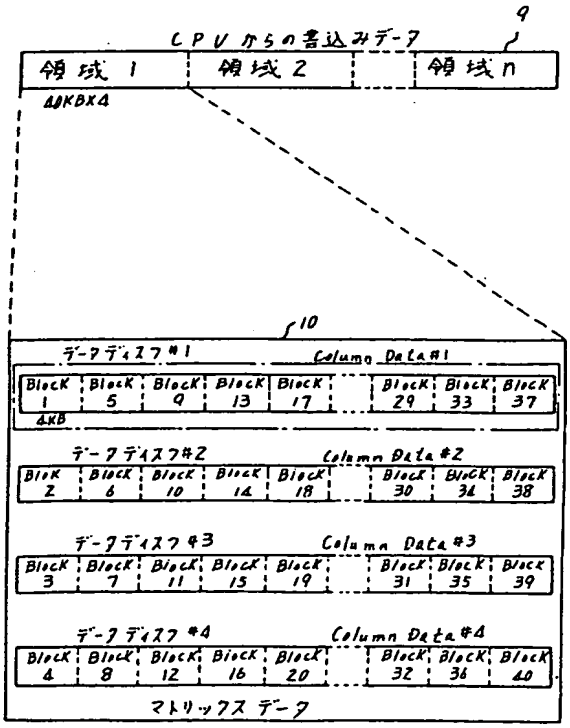
第 9 図



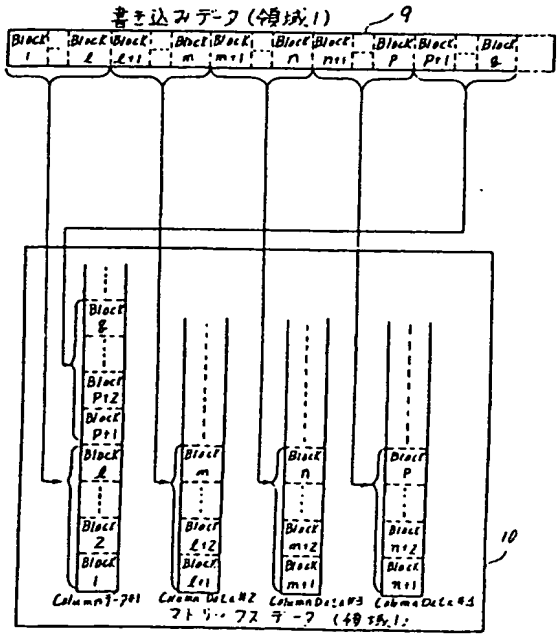
第 10 図



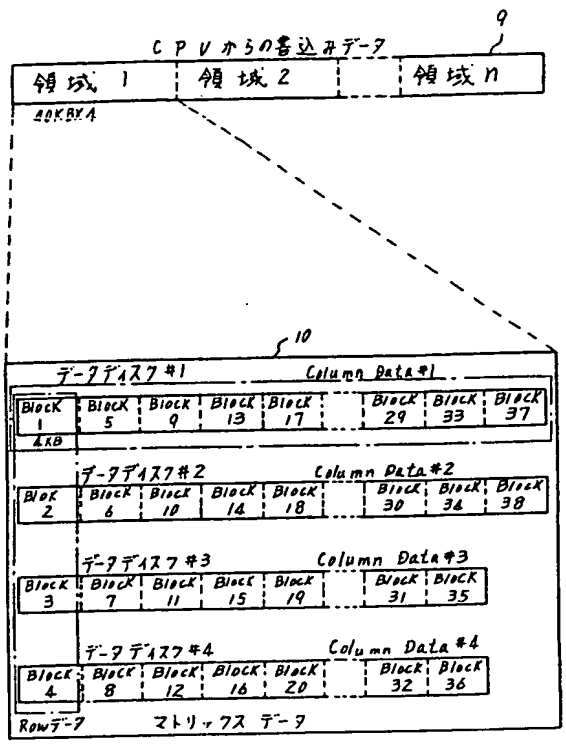
第 11 図



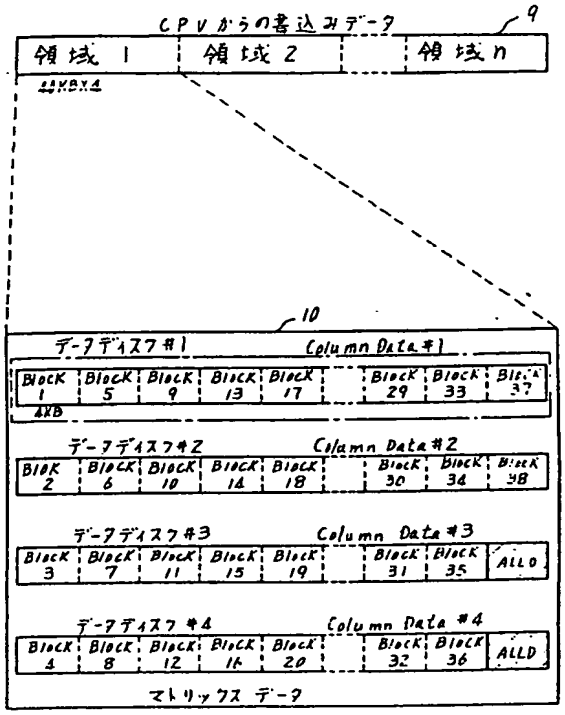
第 12 図



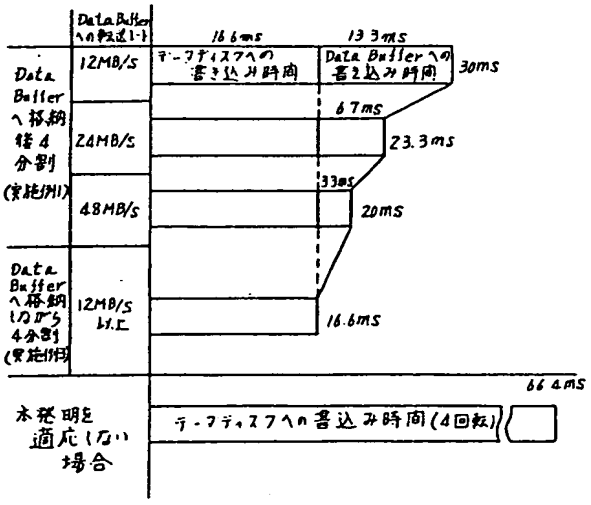
第 13 図



第 14 図

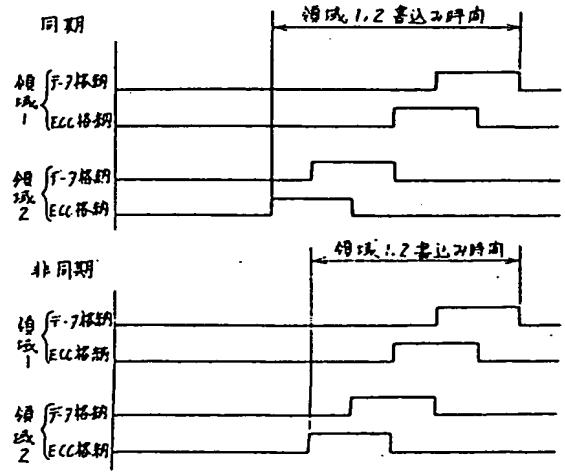


第 15 図

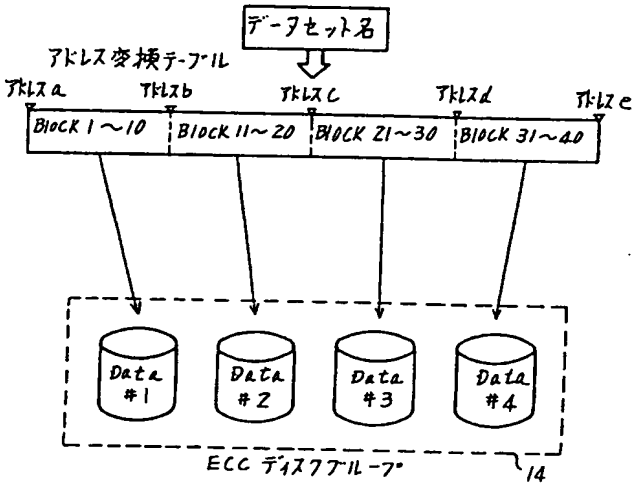


データディスクの転送レート = 3MB/s
回転数 = 3600rpm
トラック容量 = 40KB/TRK
ECCグループ内転送データ量 = 40KBX4台
= 160KB/Group

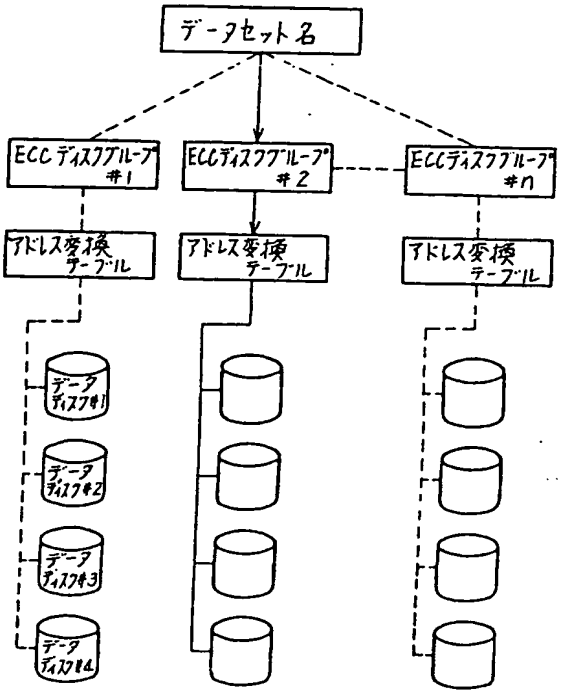
第 16 図



第 17 図



第 18 図



第 19 図